KIT SSL

Seminar

이제는 만든다: WEB 3.0

20180488 박준수

KIT SSL

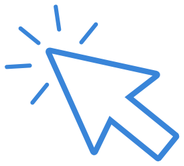
CONTENTS

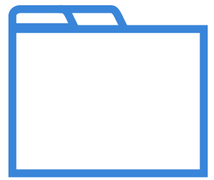
01 Recap

02 Todo list

KIT SSL

Recap





RECAP

Web3...?

웹이 발전하는 역사를 보니.. 빅테크 기업들이 모든 것을 지배한다.

중앙집중된 환경, 플랫폼 독점 문제

개인 정보 남용... 수수료도 맘대로 바꾸네?

지향점
공정해야 한다. 나눠야한다.

https://www.youtube.com/watch?v=ZUzIHjTs2dA

RECAP

그렇다면 공정. 분배 를 어떤 식으로 증명해야 하는가

블록체인!

https://www.youtube.com/watch?v=ZUzlHjTs2dA

RECAP

반드시 블록체인을 사용해야 하는가? Nope

web3의 가치를 가장 잘 뒷받침해주는 백엔드 기술일 뿐

기존 방식

신뢰하지 못한다,, 그래서 블록체인을 가지고 온 것

https://www.youtube.com/watch?v=ZUzlHjTs2dA

RECAP

블록체인에 올리면 위.변조도 안되고 되돌리기도 안되는 특성을 가지니

개인 정보 남용 안할게... 수수료도 맘대로 안 올릴게...

코드의 형식으로 블록체인 위로 올려서 자동으로 실행함

https://www.youtube.com/watch?v=ZUzlHjTs2dA

RECAP

블록체인에 올리면 위.변조도 안되고 되돌리기도 안되는 특성을 가지니

개인 정보 남용 안할게... 수수료도 맘대로 안 올릴게...

코드의 형식으로 블록체인 위로 올려서 자동으로 실행함

Smart Contract

https://www.youtube.com/watch?v=ZUzlHjTs2dA

조건이 성립되면

RECAP

Web3

공정하게 운용할 것

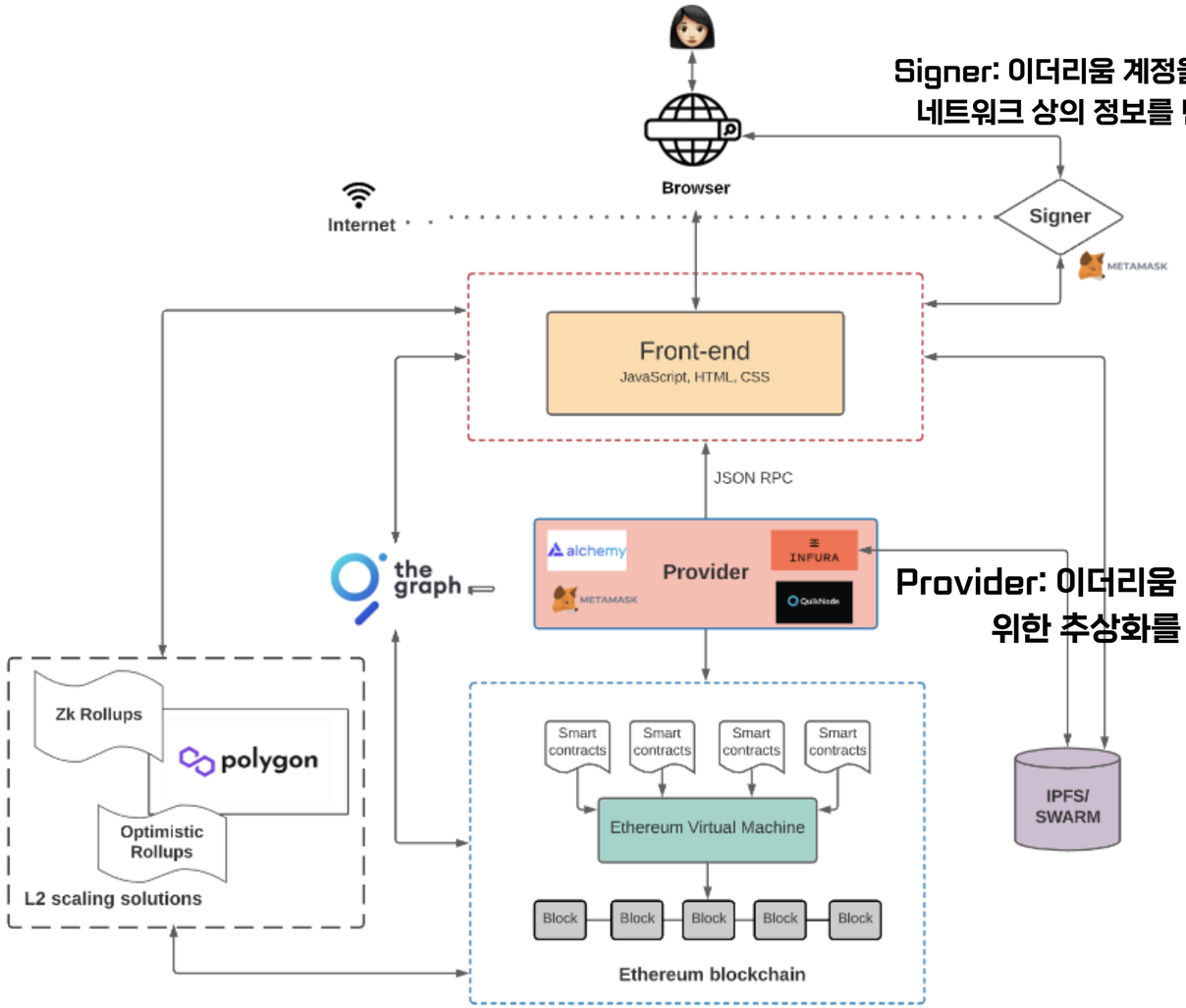
Dapp

Metaverse

Blockchain

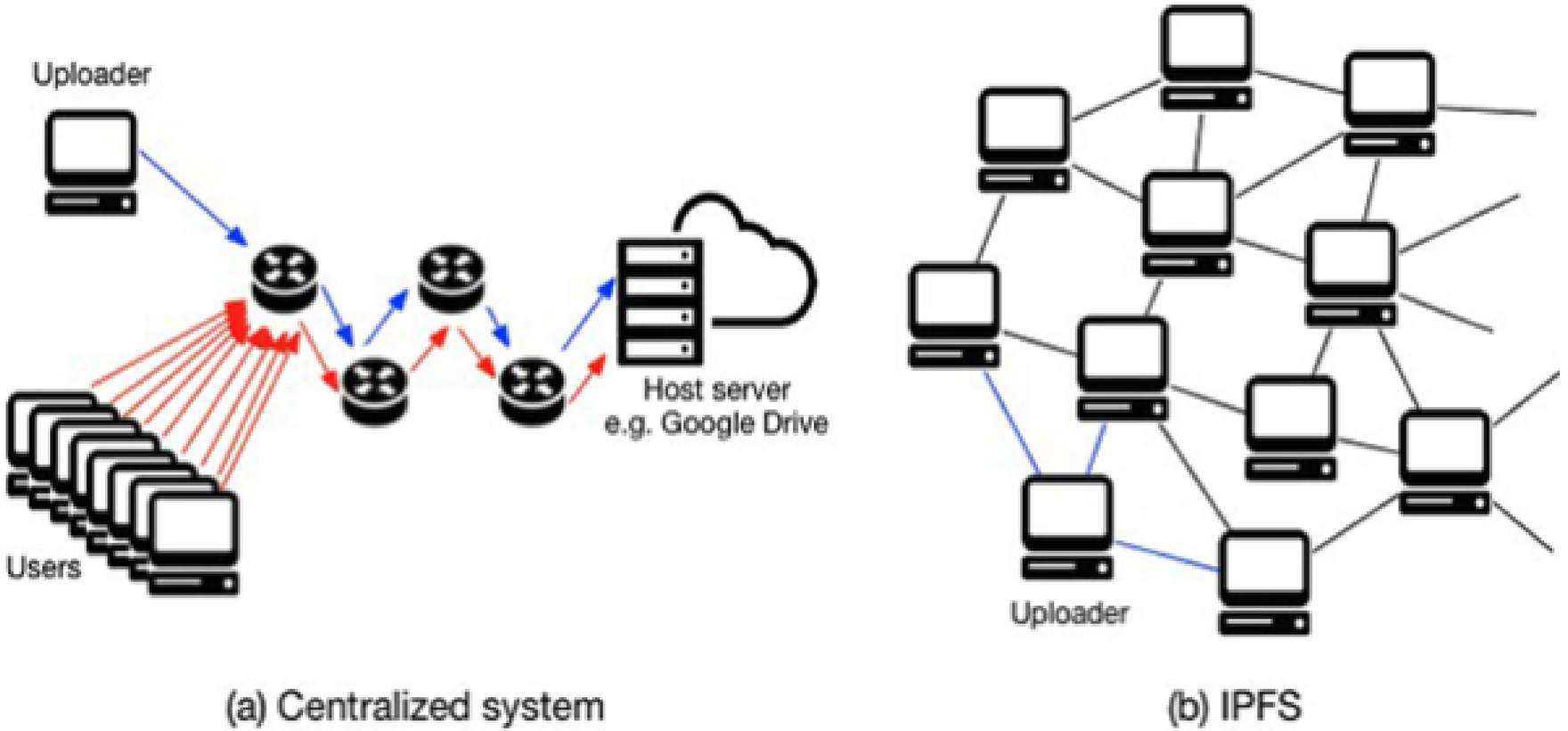
근저의 인프라

https://www.youtube.com/watch?v=ZUzlHjTs2dA



Signer: 이더리움 계정을 통해서 트랜잭션에 사인을 해서 이더리움 네트워크 상의 정보를 변경하는 트랜잭션을 실행할 수 있도록 함 (가스비 발생)

Provider: 이더리움 네트워크에 대한 연결을 위한 추상화를 제공하는 클래스



(a) Centralized system

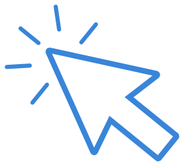
(b) IPFS

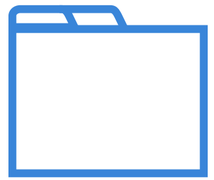
데이터를 수많은 노드에 호스팅하고 백업할 수 있는 P2P 분산 네트워크. 완전한 분산 시스템이자 네트워크인 IPFS에는 수많은 공간에 데이터가 분산되고 복제, 저장함. 한 곳의 데이터가 삭제되더라도 언제나 같은 데이터에 접근 가능

출처: students x students

KIT SSL

Todo List

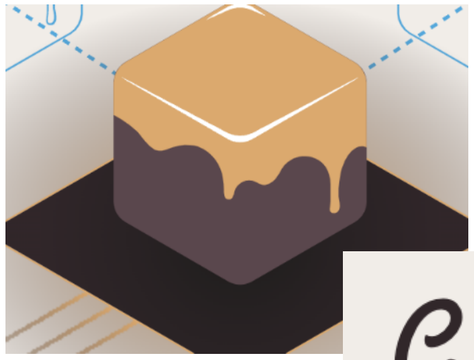




TODO LIST

TRUFFLE

Solidity로 작성된 Smart Contract
를 로컬에서 쉽게 컴파일 배포 가능하게
도와주는 프레임워크



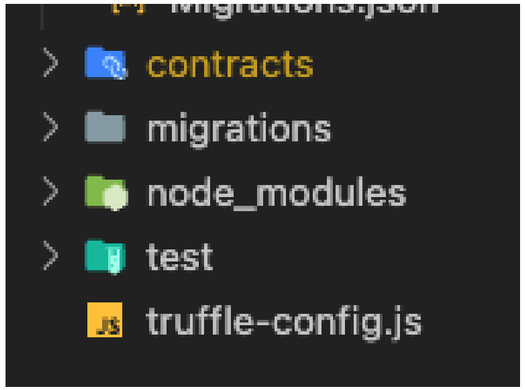
ganache-cli

Ganache

Blockchain을 쉽게 사용 가능하게함
복잡한 setting 없이 비교적 간단하게
블록체인 네트워크에 접근

Init

```
❯ cd dapp-smart
❯ truffle init

✔ Preparing to download
✔ Downloading
✔ Cleaning up temporary files
✔ Setting up box

Unbox successful. Sweet!

Commands:

  Compile:        truffle compile
  Migrate:        truffle migrate
  Test contracts: truffle test
```



contracts: 계약 파일 (.sol)

migrations: 배포시 사용되는 파일(.js)

truffle-config.js:
Truffle configration file

TODO LIST

Build

contracts > Lottery.sol

```
// SPDX-License-Identifier: MIT
pragma solidity >=0.4.22 <0.9.0;

contract Lottery{

}
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER TRUFFLE

```
❯ truffle compile
Compiling ./contracts/Lottery.sol...
Compiling ./contracts/Migrations.sol...
Writing artifacts to ./build/contracts
```

build / contracts
- Lottery.json
- Migrations.json

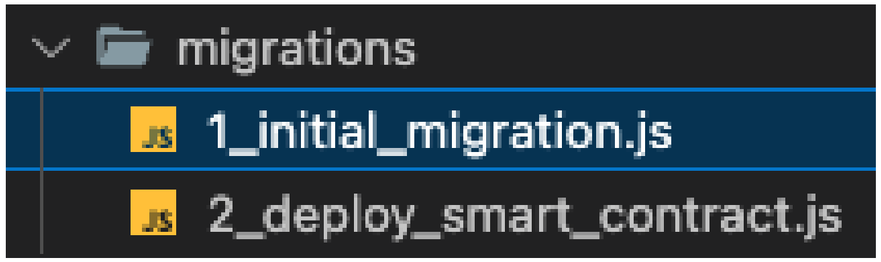
```
{
  "contractName": "Migrations",
  "abi": [⋯
  ],
  "bytecode": "0x608060405233600080610100 0a81548173ffffffffffffffffffffffffffffffffffffffff0219169083
  "deployedBytecode": "0x6080604052600436106100575760003 57c0100000000000000000000000000000000000000000
  "sourceMap": "66:352:1:-;;;113:10;90:33;;;;;;;;;;;;;;;;;;;66:352;8:9:-1;5:2;;;30:1;27;20:12;5:2;66
  "deployedSourceMap": "66:352:1:-;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;127:36;;8:9:-1;5:2;;;30:1;27;20:1
  "source": "// SPDX-License-Identifier: MIT\npragma solidity >=0.4.22 <0.9.0;\n\ncontract Migrations
  "sourcePath": "/Users/junsu/Desktop/dapp-smart/contracts/Migrations.sol",
  "ast": {
    "absolutePath": "/Users/junsu/Desktop/dapp-smart/contracts/Migrations.sol",
    "exportedSymbols": {
      "Migrations": [
        35
      ]
    },
    "id": 36,
    "nodeType": "SourceUnit",
    "nodes": [
      {
        "id": 4,
        "literals": [
```

abi : 외부에서 접근할때 이 계약에서는 어떤 함구에 접근이 가능한지, 파라미터, 리턴값이 뭔지 확인 가능

바이트 코드 : 실제 블록체인 네트워크 위로 배포될 때 사용되는 코드

컴파일을 하게되면 json의 형태로 결과가 저장됨

Migrations



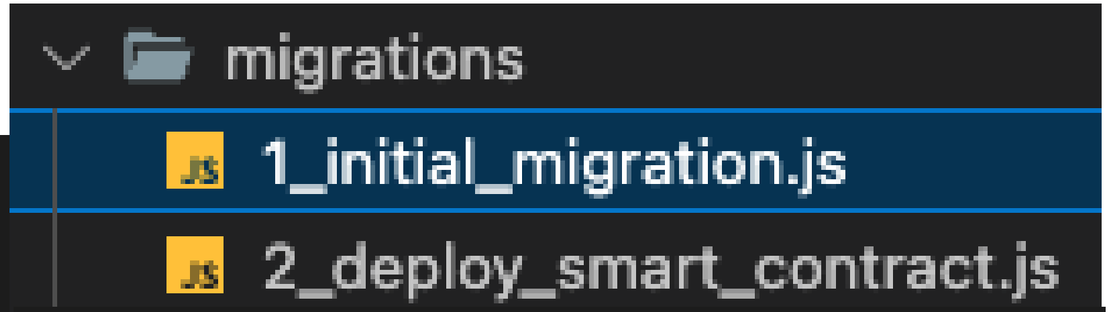

```
const Migrations = artifacts.require("Migrations");

module.exports = function(deployer) {
  deployer.deploy(Migrations);
};
```

build 안의 Migration의 데이터를 가져와 그 안의 바이트 코드를 deployer가 배포함

블록체인 네트워크 위에 컨트랙트를 배포하기 위해서는 나의 주소가 필요함
truffle-config.js에 내 주소를 세팅 하면 deployer로 매핑됨

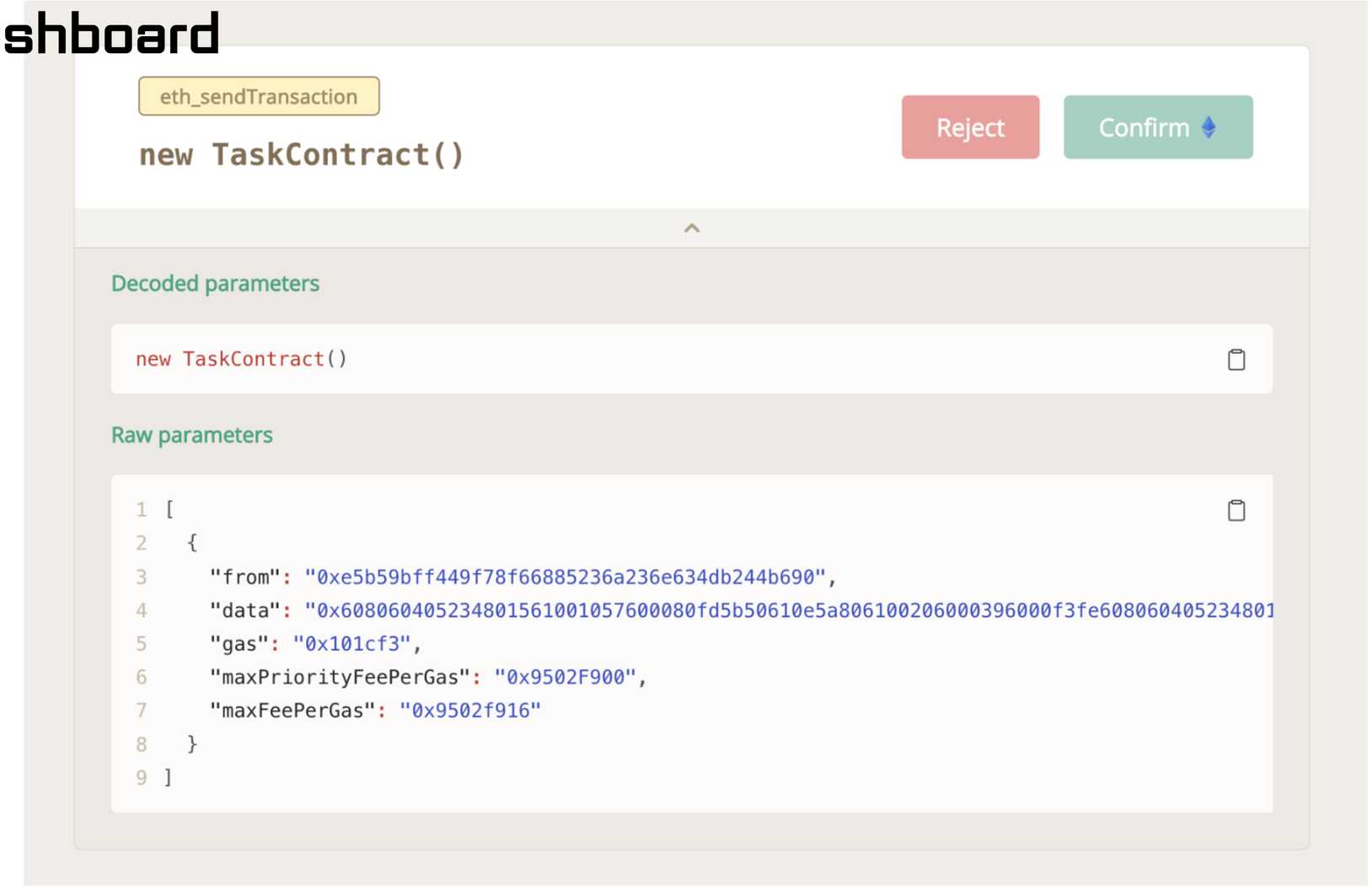
Migrations

```
❯ truffle migrate
⚠  Important ⚠
If you're using an HDWalletProvider, it must be Web3 1.0 enabled or your migration will hang.


Starting migrations...
======================
> Network name:    'development'
> Network id:      1669198192245
> Block gas limit: 30000000


1_initial_migration.js
======================

   Deploying 'Migrations'
   ----------------------
   > transaction hash:    0x0b2287aa5388b2f32f9736f5360b44a54e532d19e32d4ea7199d8b9edf167074
   > Blocks: 0            Seconds: 0
   > contract address:    0x244a6672EbDc065E9C5FF1b684720a49e2065724
   > account:             0xB0d303C3cAB0b3db314052cFd5ECeA2c46897e55
   > balance:             999.99584198
   > gas used:            207901
   > gas price:           20 gwei
   > value sent:          0 ETH
   > total cost:          0.00415802 ETH

   > Saving artifacts
   -------------------------------------
   > Total cost:          0.00415802 ETH
```




```
2_deploy_smart_contract.js
==========================

   Deploying 'Lottery'
   -------------------
   > transaction hash:    0x1fad47163177ec27e6360e3b5784fdc0a41a966cd5573ee8421bfd23b5eca1aa
   > Blocks: 0            Seconds: 0
   > contract address:    0x801EAaA59D58c7E8C13C85325fEc7EE685229951
   > account:             0xB0d303C3cAB0b3db314052cFd5ECeA2c46897e55
   > balance:             999.99454362
   > gas used:            64918
   > gas price:           20 gwei
   > value sent:          0 ETH
   > total cost:          0.00129836 ETH

   > Saving artifacts
   -------------------------------------
   > Total cost:          0.00129836 ETH


Summary
=======
> Total deployments:   2
> Final cost:          0.00545638 ETH
```

truffle dashboard



Contracts

event - 블록체인 네트워크의 블록에 특정값을 기록하는 것

```
// SPDX-License-Identifier: MIT
pragma solidity 0.8.17;

contract TaskContract {
  event AddTask(address recipient, uint taskId);
  event DeleteTask(uint taskId, bool isDeleted);

  struct Task{
    uint id;
    string taskText;
    bool isDeleted;
  }

  Task[] private tasks;
  mapping(uint256 => address) taskToOwner;

  function addTask(string memory taskText, bool isDeleted) external{…
  }

  function getMyTasks() external view returns(Task[] memory) {…
  }

  function deleteTask(uint taskId, bool isDeleted) external{…
  }
}
```

변수를 저장할 수 있는 Storage와 Memory 라는 공간이 존재
Memory는 임시적으로 저장되는 변수로 함수의 외부 호출이 일어날 때마다 초기화

external : 오직 계약 밖에서만 접근 가능

Contracts

emit - 이벤트를 출력

msg.sender -스마트컨트랙과 상호 작용하는 주체

```
function addTask(string memory taskText, bool isDeleted) external{
  uint taskId = tasks.length;
  tasks.push(Task(taskId, taskText, isDeleted));
  taskToOwner[taskId] = msg.sender;
  emit AddTask(msg.sender, taskId);
}
```

```
function deleteTask(uint taskId, bool isDeleted) external{
  if(taskToOwner[taskId] == msg.sender){
    tasks[taskId].isDeleted = isDeleted;
    emit DeleteTask(taskId, isDeleted);
  }
}
```

Contracts emit - 이벤트를 출력

```
function getMyTasks() external view returns(Task[] memory) {
  Task[] memory temp = new Task[](tasks.length);
  uint cnt = 0;

  for(uint i = 0; i<tasks.length;i++){
    if(taskToOwner[i] == msg.sender && tasks[i].isDeleted == false){
      temp[cnt] = tasks[i];
      cnt++;
    }
  }
  Task[] memory result = new Task[](cnt);
  for(uint i = 0; i<cnt ; i++){
    result[i] = temp[i];
  }
  return result;
}
```

view : function 밖의 변수들을 읽을수 있으나 변경 불가능

Home

```
export default function Home() {
    const [correctNetwork, setCorrectNetwork] = useState(false)
    const [isUserLoggedIn, setIsUserLoggedIn] = useState(false)
    const [currentAccount, setCurrentAccount] = useState("")
    const [input, setInput] = useState("")
    const [tasks, setTasks] = useState([])

    useEffect(() => {…
    }, [])

    const connectWallet = async () => {…
    }

    const getAllTasks = async () => {…
    }

    const addTask = async (e) => {…
    }

    const deleteTask = (key) => async () => {…
    }

    return (…
    )
}
```

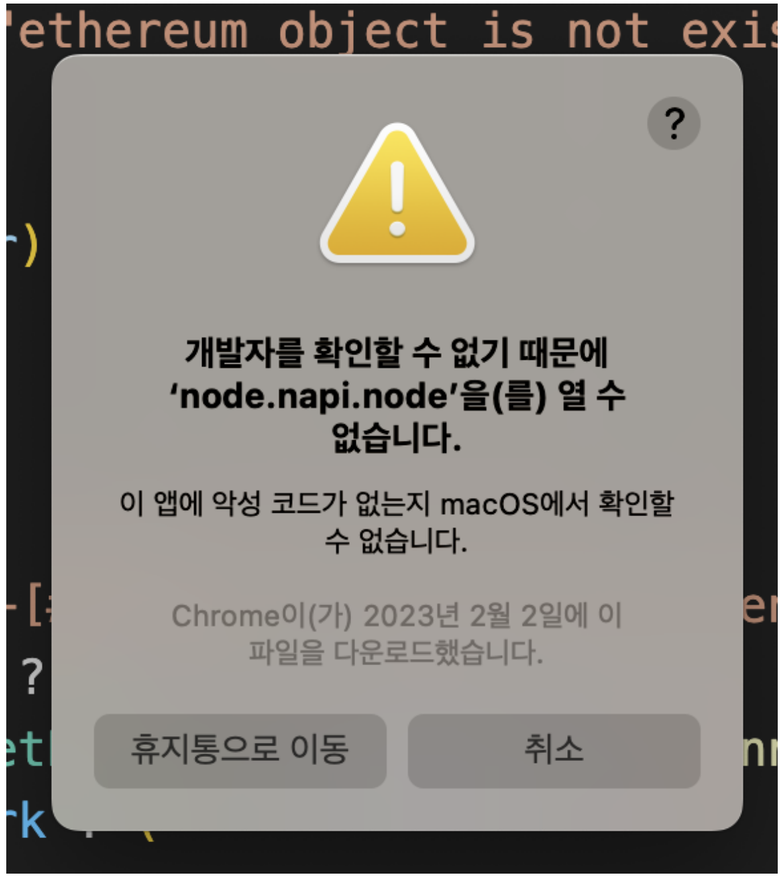
TODO LIST

```
const connectWallet = async () => {
    console.log(TaskContractAddress)
    try {
        const { ethereum } = window
        if (!ethereum) {
            console.log("Metamask not Dectected")
            return
        }
        let chainId = await ethereum.request({ method: "eth_chainId" })
        console.log("Connected to chain: ", chainId)

        const goerliChainId = "0x5"
        if (chainId !== goerliChainId) {
            alert("You are not in Goerli")
            setCorrectNetwork(false)
            return
        } else {
            setCorrectNetwork(true)
        }

        const accounts = await ethereum.request({
            method: "eth_requestAccounts",
        })
        console.log("Found Account ", accounts[0])
        setIsUserLoggedIn(true)
        setCurrentAccount(accounts[0])
    } catch (error) {
        console.log(error)
    }
}
```

```
const addTask = async (e) => {
    e.preventDefault()
    let task = {
        taskText: input,
        isDeleted: false,
    }
    try {
        const { ethereum } = window
        if (ethereum) {
            const provider = new ethers.providers.Web3Provider(ethereum)
            const signer = provider.getSigner()
            const TaskContract = new ethers.Contract(
                TaskContractAddress,
                TaskAbi.abi,
                signer
            )
            TaskContract.addTask(task.taskText, task.isDeleted)
                .then((res) => {
                    setTasks([...tasks, task])
                    console.log(tasks)
                    console.log("Added tasks")
                })
                .catch((err) => {
                    console.log(err)
                })
        } else { console.log("ethereum object is not exist..3") }
    } catch (error) {
        console.log(error)
    }
    setInput("")
}
```

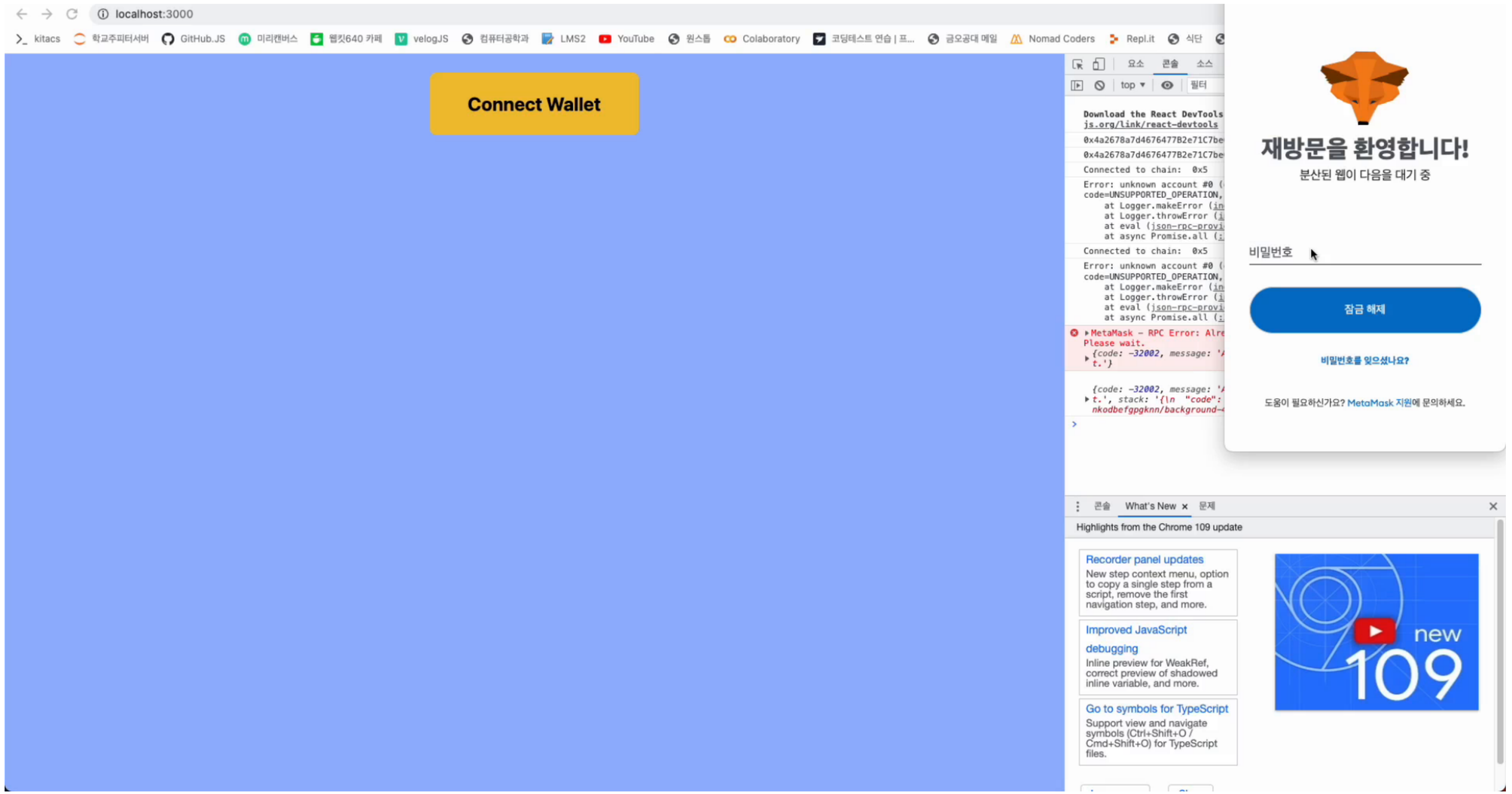
TODO LIST

```
return (
    <div className="bg-[#97b5fe] h-screen w-screen flex justify-center py-6">
        {!isUserLoggedIn ? (
            <ConnectWalletButton connectWallet={connectWallet} />
        ) : correctNetwork ? (
            <TodoList
                tasks={tasks}
                input={input}
                setInput={setInput}
                addTask={addTask}
            />
        ) : (
            <WrongNetworkMessage />
        )}
    </div>
)
```



주인이 만든 것도 못알아보고...

TODO LIST


localhost:3000
Connect Wallet
재방문을 환영합니다!
분산된 웹이 다음을 대기 중
비밀번호
잠금 해제
비밀번호를 잊으셨나요?
도움이 필요하신가요? MetaMask 지원에 문의하세요.
What's New
Highlights from the Chrome 109 update
Recorder panel updates
New step context menu, option to copy a single step from a script, remove the first navigation step, and more.
Improved JavaScript debugging
Inline preview for WeakRef, correct preview of shadowed inline variable, and more.
Go to symbols for TypeScript
Support view and navigate symbols (Ctrl+Shift+O / Cmd+Shift+O) for TypeScript files.
new
109

TODO:

1. 삭제 기능 구현

2. ipfs 연결 후 데이터 저장하기

TODO LIST

쿠키

https://gateway.ipfscdn.io/ipfs//QmU3CzXXZ4Mcfj6wWFDFWKwbX8NL2fxry56tqwuEPa1m9u/marp_cookie.html

출처

[1] https://www.youtube.com/watch?v=awQTDVvYyjI
[2] https://www.youtube.com/watch?v=ZUzIHjTs2dA
[3] https://velog.io/@wrjang96/provider-web3-react-ethers.js-%EA%B0%9C%EB%85%90-%EC%A0%95%EB%A6%AC
[4] https://www.lgcns.com/blog/it-trend/31193/

EOF